CLAIMS

1. A method of performing memory mapped input output operations to an alternate address space comprising:

establishing a first instruction directed to a first memory mapped input output alternate address space associated with an adapter to store data in accordance with a definition of a z/Architecture;

establishing a second instruction directed to said first memory mapped input output alternate address space associated with an adapter to load data in accordance with a definition of a z/Architecture;

allocating at least one of a real resource and a virtual resource associated with said first alternate address space to a process;

ensuring that said selected process corresponds with said process to which said resource is allocated; and

wherein said process issues at least one of said first instruction and said second instruction and thereby causes execution of at least one of said store and load with said first alternate address space.

2. The method of Claim 1 further including an error storage-area associated with an alternate address space that is further associated with an error storage area, storage-area type, fetched by execution of said second instruction.

3. The method of Claim 1 further including virtualization of a resource of said adapter to a second-level guest process.

4. The method of Claim 3 wherein said virtualization of a resource is accomplished and distinguished from a real resource by partitioning a range of resource identifiers into a plurality of portions;

wherein at least one portion corresponds to a virtual resource; and

wherein when at least one of said first instruction and said second instruction specifies a resource identifier corresponding to said at least one portion, the guest issuing said instruction exits, and an underlying host program resumes execution in order to emulate said at least one of said first instruction and said second instruction originally issued by the guest.

5. The method of claim 3 wherein said virtualization provides direct access to at least one of a real resource and a virtual resource of an adapter by a problem-state second-level guest process

6. The method of claim 5 wherein said access is accomplished without involvement from a kernel of said guest operating system; and permits said process operating in a problem-state maximum efficiency in performing the primary input output capabilities provided by said adapter and the associated resources allocated to said process.

7. The method of Claim 3 further including separating another process operating under said operating system; wherein said separating is established on a per-resource basis during said allocating and is enforced during execution of at least one of said first and said second instructions.

8. The method of Claim 1wherein said first alternate address space is not a portion of the main address space from which said process is executing.

9. The method of Claim 1 wherein said process issuing said at least one of said first instruction and said second instruction and thereby causes execution of at least one of said store and load with said first alternate address space operates in a problem state of a machine

10. The method of Claim 1 wherein said problem state corresponds to a least privileged execution state in said z/Architecture

11. The method of Claim 1 wherein said first alternate address space is associated with an adapter and alleviates use of a main address space of said process or of another adapter.

12. The method of Claim 1 wherein at least one of said first instruction and said second instruction is executed without supervisory state intervention.

13. The method of Claim 1 wherein said first instruction and said second instruction are semiprivileged instructions that may be executed in problem state, wherein ownership of a specified resource of a specified adapter determines a privilege required for execution of said semiprivileged instructions.

14. The method of Claim 1 further including a second alternate address space associated with a second adapter.

15. The method of Claim 14 wherein a storage location in said first alternate address space maps to a different address than the same location in said second alternate address space.

16. The method of Claim 1 wherein said adapter includes address spaces as partitions of said alternate address space.

17. The method of Claim 1 wherein an address space is governed by at least one of a resource type and storage area types associated with an adapter.

18. A storage medium encoded with a machine-readable computer program code, said code including instructions for causing a computer to implement a method of performing memory mapped input output operations to an alternate address space, the method comprising:

establishing a first instruction directed to a first memory mapped input output alternate address space associated with an adapter to store data in accordance with a definition of a z/Architecture;

establishing a second instruction directed to said first memory mapped input output alternate address space associated with an adapter to load data in accordance with a definition of a z/Architecture;

allocating at least one of a real resource and a virtual resource associated with said first alternate address space to a process;

ensuring that said selected process corresponds with said process to which said resource is allocated; and

wherein said process issues at least one of said first instruction and said second instruction and thereby causes execution of at least one of said store and load with said first alternate address space.

19. A system for performing memory mapped input output operations to an alternate address space comprising:

a means for establishing a first instruction directed to a first memory mapped input output alternate address space associated with an adapter to store data in accordance with a definition of a z/Architecture;

a means for establishing a second instruction directed to said first memory mapped input output alternate address space associated with an adapter to load data in accordance with a definition of a z/Architecture;

a means for allocating at least one of a real resource and a virtual resource associated with said first alternate address space to a process;

a means for ensuring that said selected process corresponds with said process to which said resource is allocated; and

wherein said process issues at least one of said first instruction and said second instruction and thereby causes execution of at least one of said store and load with said first alternate address space.